IDENTIFYING STUCK-AT-FAULTS OF FULL ADDER USING FPGA AS TESTING DEVICE

ADDENKI NAVEENA
ELECTRONICS AND COMMUNICATION DEPARTMENT, NALLA NARASIMHA GROUP OF INSTITUTIONS, HYDERABAD GHATKESAR, INDIA, a.naveena28@gmail.com

DURGAM SHIVA
ELECTRONICS AND COMMUNICATION DEPARTMENT, NALLA NARASIMHA GROUP OF INSTITUTIONS, HYDERABAD, GHATKESAR, INDIA, durgamshiva29@gmail.com

CHINTHALAPALLY SAHITHI
ELECTRONICS AND COMMUNICATION DEPARTMENT , NALLA NARASIMHA GROUP OF INSTITUTIONS, HYDERABAD, GHATKESAR, INDIA, Sahithireddy1108@gmail.com

ABHI RAM SAMALLA
ELECTRONICS AND COMMUNICATION DEPARTMENT , NALLA NARASIMHA GROUP OF INSTITUTIONS, HYDERABAD, GHATKESAR, INDIA, abhiramsamalla@gmail.com

ALAMPALLY ARAVIND, Assistant Professor
ELECTRONICS AND COMMUNICATION DEPARTMENT, NALLA NARASIMHA GROUP OF INSTITUTIONS, HYDERABAD, GHATKESAR, INDIA, aravindalampally89@gmail.com

ABSTRACT: Advanced CMOS devices are a critical issue due to the shrinking process of the size of the transistor. The number of transistors packed in an IC increases with the latest technology. Manufacturing defects in the chip happen due to the interconnection of wires. This leads to unexpected outputs. The process of testing confirms that the chip is fault free. This work discusses the number of test vectors needed to find the faults present in a full adder. It is achieved with the help of a field programmable gate array (FPGA). An adder circuit is realized in FPGA. Test vectors are also implemented in FPGA which is fed as inputs to the full adder. Thus generated outputs are tested with the expected outputs to identify the faulty position(s) in the full adder. Zynq-7000 FPGA chip was used to realize the testing.

INDEX TERMS: FPGA, Verilog, Xilinx, Stuck-At-Faults, Fault models.

I.INTRODUCTION:

The evolution of integrated circuits started with the introduction of the microprocessor. Proper functioning of a chip is tested before it is manufactured. Various types of the verification involved in the process of chip testing are IP verification, RTL verification, timing verification, etc. Testing becomes a headache in the process of IC fabrication as it consumes 80% of the manufacturing time. A faulty product is generated due to in complete specification problem, incorrect design, faulty fabrication process, and wrong test method. Therefore, it is necessary to identify the locations at which the faults are present. Add or simulate a fault to any one or more lines of the circuit for testing a chip. There are various types of fault occur in the circuit, they are Stuck-at-Fault, Delay Fault, Transient Fault, Bridging Fault, and so on. These types of faults may occur either on the input or output side.A full adder is a fundamental building block in digital circuit design, and it's used to add three binary numbers: A, B, and a carry-in (Cin). It produces a sum and a carry-out (Cout) as outputs.

II.LITERATURE SURVEY:

Basic parts of digital circuits, full adders are frequently used to add binary numbers. When designing digital systems, it is imperative to ensure their reliability. To evaluate their robustness, fault testing is one method. This review of the literature investigates the use of Field-Programmable Gate Arrays (FPGAs) as testing devices to implement stuck-at fault testing in full adders. A full adder is a combinational logic circuit that generates a sum (S) and a carry-out (Cout) by adding three binary inputs: A, B, and a carry-in (Cin). XOR, AND, and OR gates are commonly used in its implementation. One popular fault model for digital circuits is the stuck-at fault model. This model assumes that a specific line becomes stuck at a logic high or low due to a fault. FPGAs are flexible electronics that can be set up to perform as a variety of digital circuits. They are perfect for fault testing because they are reprogrammable. Using FPGAs to implement stuck-at fault testing in full adders entails setting up the FPGA to emulate the full adder's typical behavior before adding faults to test the FPGA's fault detection capabilities. Several research studies have focused on the implementation of stuck-at fault testing in full adders

using FPGAs. Some notable works include: FPGA-based testing of full adders for stuck-at faults is a promising approach for assessing the reliability and robustness of these fundamental building blocks in digital systems. Research in this area has made significant strides, but there are still opportunities for improvement and further exploration. The application of FPGA-based testing methodologies can contribute to the development of more reliable digital systems. Despite the advantages of FPGA-based testing, there are challenges such as fault localization, test vector generation, and the need for specialized hardware.

1. J. Zhou, Z. Cao, X. Dong, and A. V. Vasilakos, "Security and privacy for cloud-based IOT: Challenges," .The Internet of Things (IOT) is becoming the next Internet-related revolution. It allows billions of devices to be connected and communicate with each other to share information that improves the quality of our daily lives. On the other hand, Cloud Computing provides on-demand, convenient and scalable network access which makes it possible to share computing resources, indeed, this, in turn, enables dynamic data integration from various data sources. There are many issues standing in the way of the successful implementation of both Cloud and IOT. The integration of Cloud Computing with the IOT is the most effective way on which to overcome these issues. The vast number of resources available on the Cloud can be extremely beneficial for the IOT, while the Cloud can gain more publicity to improve its limitations with real world objects in a more dynamic and distributed manner. This paper provides an overview of the integration of the Cloud into the IOT by highlighting the integration benefits and implementation challenges. Discussion will also focus on the architecture of the resultant Cloud-based IOT paradigm and its new applications scenarios. Finally, open issues and future research directions are also suggested.

2. E. Zenner, "Cryptanalysis of LFSR-based pseudorandom generators—A survey," Pseudorandom bit generator (PRBG) is an essential component for securing data during transmission and storage in various cryptography applications. Among popular existing PRBG methods such as linear feedback shift register (LFSR), linear congruential generator (LCG), coupled LCG (CLCG), and dual-coupled LCG (dual-CLCG), the latter proves to be more secure. This method relies on the inequality comparisons that lead to generating pseudorandom bit at a non-uniform time interval. Hence, a new architecture of the existing dual CLCG method is developed that generates pseudo-random bit at uniform clock rate. However, this architecture experiences several drawbacks such as excessive memory usage and high-initial clock latency, and fails to achieve the maximum length sequence. Therefore, a new PRBG method called as "modified dual-CLCG" and it's very large-scale integration (VLSI) architecture are proposed in this paper to mitigate the aforesaid problems. The novel contribution of the proposed PRBG method is to generate pseudorandom bit at uniform clock rate with one initial clock delay and minimum hardware complexity. Moreover, the proposed PRBG method passes all the 15 benchmark tests of NIST standard and achieves the maximal period of 2^n. The proposed architecture is implemented using Verilog-HDL and prototyped on the commercially available FPGA device.

III.METHODOLOGY:

3.1.FAULT TOLERANT TESTING ALGORITHM FOR FULL ADDER:

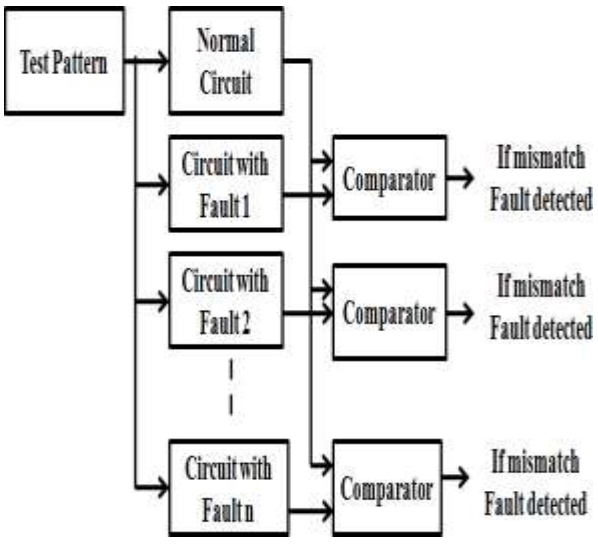

Fig 3.1: Basic Principle of Fault Simulation.

The evolution of integrated circuits started with the introduction of the microprocessor. Proper functioning of a chip is tested before it is manufactured. Various types of the verification involved in the process of chip testing are IP verification, RTL verification, timing verification, etc., Testing becomes a headache in the process of IC fabrication as it consumes 80% of the manufacturing time. A faulty product is generated due to incomplete specification problem, incorrect design, faulty fabrication process, and wrong test method. Therefore, it is necessary to identify the locations at which the faults are present. Add or simulate a fault to any one or more lines of the circuit for testing a chip. There are various types of fault occur in the circuit, they are Stuck-at-Fault, Delay Fault, Transient Fault, Bridging Fault, and so on. These types of faults may occur either on the input or output side of the circuit.The two types of stuck-at-faults are Stuck-at-1 (SA1) and Stuck-at-0 (SA0). Stuck-at

fault identifies the faults that are available in an integrated circuit. It can occur either at the input to the gate or at the output of the gate. When the value of the line or cell is always '0', it is identified as a stuck-at-0 fault (SA0). When the value of the line or cell is always '1', it is identified as a stuck-at-1 fault (SA1). This stuck-at fault model also helps to identify the manufacturing defect of a logic gate itself. Faults in the chip are identified through fault simulation . A typical testing process involves the generation of test vectors or test patterns for the circuit which is to be tested. These test vectors are applied as inputs to the circuit. Outputs of the circuit are verified with the expected outputs. Faults identification are decided based on the result of the comparison. The basic principle of fault simulation is shown . Testing is normally done using a computer with the device under test (DUT). An algorithm developed using a high-level language generates test vectors. These test vectors are in such a way that it provides all the possible combination of inputs that are applied to the DUT. A suitable data transfer medium transfers the test vectors to the DUT and receives the outputs generated by it. Now, the algorithm analyzes the outputs and identifies the presence of the fault(s). The testing time depends on the number of test vectors [4-5]. Whereas, the number of test vectors is based on the possible combinations of inputs that are applied to the DUT. Hence, an optimistic algorithm always tries to reduce the total number of test vectors and hence the testing time. This work has attempted to establish testing using a field programmable gate array (FPGA). FPGA is known for its unique advantage called concurrent execution. This motivates the researchers to develop applications using FPGA. It is flexible and re- programmable. Latest FPGAs have in-built features like DDR, DSP, MAC etc., It supports the development of applications in the fields of image processing, instrumentation, medical, signal processig.

3.2.LINEAR CONGRUENTIAL GENERATOR:

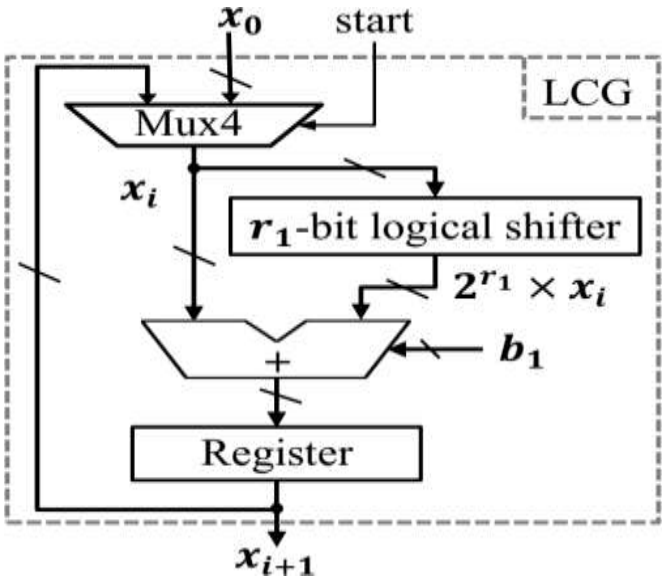

Fig.3.2: Architecture of the linear congruential generator.

These pseudo random numbers are generated by **linear congruential generators** (**LCG**). The principle of an LCG is quite simple: a new pseudo random number is generated on the basis of the previous random number by adding a certain offset and wrapping the result if it exceeds In fact, any higher level programming language offers at least one form of random number generator. The generation of random numbers, however, is not an easy task for a computer, since the computer is a deterministic machine with no built-in randomness. Thus it is impossible to create true random numbers without any additional hardware. What can be done, is to create **pseudo random numbers** which behave almost like random numbers but which are repeated after a fixed (mostly quite long) period. a certain limit. The process can be denoted by the following equation: The linear congruential method produces a sequence of integers $X_0, X_1, X_2 \ldots$ between zero and *m-1* according to the following recursive relations

$$x_{i+1} = a_1 \times x_i + b_1 \bmod 2^n \ldots\ldots(1)$$

Following are the necessary conditions to get the maximum period, b_1 is relatively prime with $2^n(m).a_1$ must Here, a1, b1 are the constant parameters; x_i is the initial seed, be divisible by 4.

LCG method is developed to generate pseudorandom bit at an equal interval of time for encrypting continuous data stream in the stream cipher.The architecture is designed with two comparators, four LCG blocks, one controller unit and memory (flip-flops) as shown in Fig. 1. The LCG is the basic functional block in the dual-CLCG architecture that involves multiplication and addition processes to compute n-bit binary random number on every clock cycle. The multiplication in the LCG equation can be implemented with shift operation, when a is considered as (2^r+1). Here, r is a positive integer, $1 < r < 2^n$. Therefore, for the efficient computation of xi+1, the equation (1) can be rewritten as,

$Xi+1 = (a1 \times xi + b1) \bmod 2^n = [(2^{rl} + 1)\, xi + b1] \bmod 2^n = [(2^{rl} \times xi) + xi + b1] \bmod 2^n$

The architecture of LCG shown in Fig. 3 is implemented with a 3-operand modulo 2n adder, 2 × 1 n-bit multiplexer and n-bit register. LCG generates a random n-bit binary equivalent to integer number in each clock cycle. Other three LCG equations can also be mapped to the corresponding architecture similar to the LCG equation (1).

The two types of stuck-at-faults are Stuck-at-1 (SA1) and Stuck-at-0 (SA0). Stuck-at fault identifies the faults that are available in an integrated circuit. It can occur eitherat the input to the gate or at the output of the gate. When the value of the line or cell is always '0', it is identified as astuck-at-0 fault (SA0). When the value of the line or cell isalways '1', it is identified as a stuck-at-1 fault (SA1). This stuck-at fault model also helps to identify the manufacturing defect of a logic gate itself. Faults in the chip are identified through fault simulation . A typical testing process involves the generation of test vectors or test patterns for the circuit which is to be tested. These test vectors are applied as inputs to the circuit. Outputs of the circuit are verified with the expected outputs. Faults identification are decided based on the result of the comparison.

Different kinds of algorithms used for fault simulation are serial fault simulation, parallel fault simulation, concurrent fault simulation, and deductive fault simulation.This work concentrates on parallel fault simulation. Parallel fault simulation is the most effective technique when a circuit is designed using logic gates and modeled with stuck-at- faults. The testing flow diagram shown in Fig. . The flow will continue till find a required test vectors.

3.3. TESTING FLOW DIAGRAM:

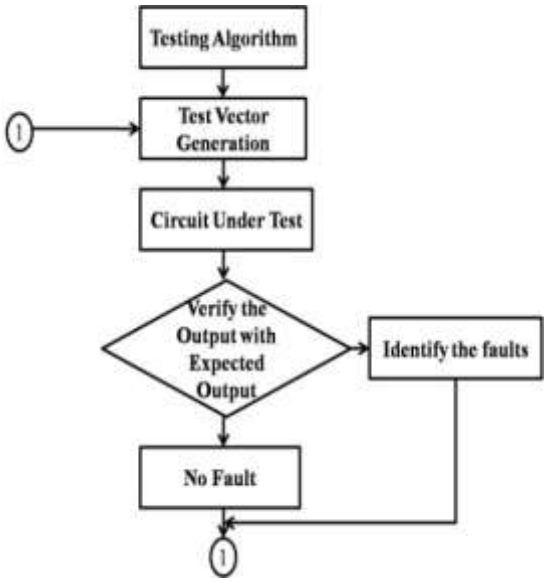

Fig.3.3**:** Testing Flow Diagram.

This work uses the application and testing algorithm in the same chip. It can be re- configured at any point of time with the help of hardware description languages (HDL). The bit file generated after synthesis is implemented into the Xilinx. Xilinx realizes the HDL coding in the form of logic gates.

Path sensitizing method of testing has been a powerful approach for test generation of any combinational circuit. Path Sensitization is based on the assumption that the failure mechanism in a gate results in its inputs or outputs being stuck-at-1 or stuck-at-0. This fault is detected along a path to the circuit output and its effects are noted. This impact determines the occurrence of a fault. It also helps to identify how the fault is propagated to the output by means of a sensitized path or path from the fault through the associated gate. Realization of path sensitizing in a fulladder circuit is shown in Fig 3. Path sensitization can be done through the Boolean difference method [9-10].

The Boolean difference method is a well known mathematical concept which has found significant application in the single fault analysis of combinational logic circuits. Full adder circuit has two outputs as 'sum' and 'carry'. Consider the output 'sum' which has an input ofa, b, and c. If any one of the input of the full adder circuit has stuck-at-faults, then the output 'sum' may lead to erroneous output. Now, let line 1 has a stuck-at-0 fault. Boolean difference method helps to identify this fault**.**

3.4.RESULT:

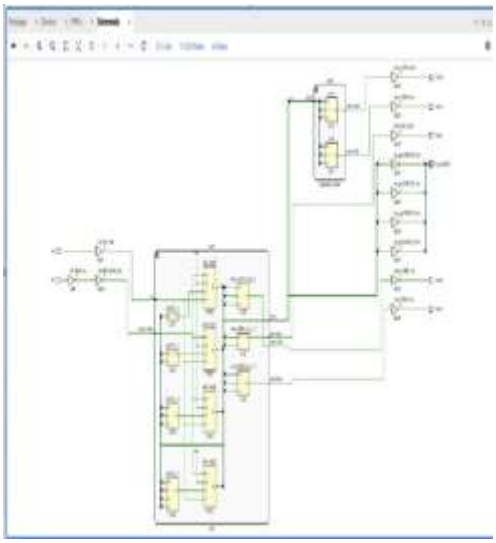

Fig 3.4.1: Proposed Method Schematic diagram

Fig 3.4.2: Proposed Method Waveforms.

INPUT	INPUT	OUTPUT	OUTPUT	OUTPUT	OUTPUT	OUTPUT	OUTPUT
clk	rst	Lcg_out[3:0]	Esum	Ecarry	tsum	tcarry	fault
1	0	0001	1	0	1	0	0
1	0	0011	0	1	0	1	0
1	0	0100	1	0	1	0	0
1	0	0110	0	1	0	0	1
1	0	0111	1	1	1	0	1
1	0	1100	1	0	1	0	1
1	0	1110	0	1	0	0	1
1	0	1111	1	1	1	0	1

Fig 3. 4.3: Truth Table of Proposed Method.

IV.CONCLUSION:

In conclusion, the successful implementation of a stuck-at fault for a full adder in a FPGA demonstrates the feasibility of this approach in simulating complex hardware faults. However, it is important to note that while the use of FPGAs provides a practical platform for implementing such simulations, there are inherent limitations to this approach. These limitations include the possibility of timing-related issues and the complexity of large-scale circuits, which can make the process of simulating and debugging these faults time-consuming and challenging.Future work in this area could focus on refining the techniques used to implement and simulate stuck-at faults in FPGA-based platforms. Additionally, researchers could explore alternative approaches to hardware fault simulation, such as using software emulation or other simulation platforms, to potentially overcome some of the limitations associated with FPGA-based simulations.Overall, this research provides valuable insights into the potential benefits and challenges of using FPGA-based platforms for simulating complex hardware faults. The successful implementation of a stuck-at fault for a full adder in a FPGA serves as a strong testament to the feasibility and potential of this approach, and highlights the need for continued exploration and development in this field.In conclusion, the successful implementation of a stuck-at fault for a full adder in a FPGA demonstrates the feasibility of this approach in simulating complex hardware faults. While there are inherent limitations to this approach, such as timing-related issues and the complexity of large-scale circuits, future work in this area could focus on refining the techniques used to implement and simulate stuck-at faults in FPGA-based platforms.

V. REFERENCES :

[1]. P. V. Reddy and K. D. Wagh, "Design of full adder fault tolerant circuit," 2010 2nd International Conference on Innovative Communication Networks, IEEE, pp.313-316, 2010.

[2]. R. Zhu and D. Chen, "Design and simulation of stuck-at faults for an n-bit ripple carry adder," IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 16, no. 6, pp. 889-901, 2008.

[3]. R. H. Grover, R. H. Jones, and A. T. Ettouf, "Formal methods in integrated circuit testing: With special emphasis on testing RAM chips," in Integrated Circuit Testing and Debugging: A Source Book, 1st ed., vol. 2, J. E. Bryant, Ed., Prentice Hall,1986.

[4]. C. I. Dekhtyar, R. B. Patil, and D. L. Tabak, "Techniques for characterizing the propagation of stuck-at faults," in Design Automation and Test in Integrated Circuits (DATAC '93), IEEE, 1993, pp. 469-472.

[5]. D. F. White and C. G. Silverman, "On the electrical and optical stability of lithography techniques," Journal of Optical Society of America B, vol. 22, no. 2, pp. 157-163, 2005.

[6]. Biernat, J., "fast issue tolerant adders," worldwide magazine crucial pc-based structures, Vol. 1, Nos. half of/three, 2010, pp.117–127.

[7]. A.Mukherjee and A. S.Dhar,"plan of a Self-Reconfigurable Adder for Fault-Tolerant VLSI engineering," 2012 worldwide Symposium on electronic device design (ISED), Kolkata, 2012, pp. 90 - 96 . doi: 10.1109/ISED.2012.21.

[8]. Somashekhar, Vikas Maheshwari, R. P. Singh, "assessment of Micro Inversion to embellish Fault Tolerance in extreme beat VLSI Circuits", global examinations mag of Engineering and innovation (IRJET), amount: 06 inconvenience: 03, web page 5041-5044, Mar 2019.